

REMARKS

Claims 1-7 have been rejected under 35 USC 103(a) as being unpatentable over Piersol, Kurt W. ("Object Oriented Spreadsheets: The Analytic Spreadsheet Package," OOPSLA '86 Proceedings, pg 385-390: Sept., 1986; herein "Piersol") in view of Levoy, Marc ("Spreadsheets for Images", Computer Graphics Proceedings, Annual Conference Series pg 139-146; 1994; herein "Levoy") and in further view of Smith (US Pat. No. 6,222,531, filed 12/10/98, herein "Smith").

In summary, combining Piersol, Levoy, and Smith is not taught, suggested, or motivated by any of these references, alone or in combination. Even if they were to be combined, the result would not be Applicant's invention because each reference fails to remedy the deficiency of the combination of the other two. Further, it would require a degree of experimentation that rises to the level of invention to combine these references and whatever addition references that would be necessary to remedy the deficiency that remains after combination thereof. Accordingly, these references fail to render Applicant's invention, as set forth in claims 1-7, obvious under 35 USC 103(a). These assertions will be now supported in detail herein below.

Regarding independent claim 1, the Examiner asserts that Piersol discloses a spreadsheet that includes cells in which objects are instantiated, but at page 385, col. 2, paragraph 2, lines 5-8, it's clear that Piersol merely discloses that "images ... have

been included in ASP cells". An **image** included in an ASP cell is NOT equivalent to a **single method object** instantiated in a cell of Applicant's invention. Moreover, a **cell** is NOT a **single method object** of Applicant's invention, as will now be explained.

Piersol states that "Each cell in an ASP spreadsheet is an object of class SpreadsheetCell. By making each cell a separate object, operations on the cells themselves (as opposed to cell values) are simplified." (page 388, col. 1, paragraph 1, lines 1-4) Thus, in Piersol, **images are stored in objects**, i.e., the **cells** of Piersol's spreadsheet, whereas in Applicant's invention, **images are stored in respective "single method objects"**, that are in turn instantiated in at least one cell of Applicant's spreadsheet.

The Examiner cites Fig. 1 of Piersol as showing an example of an object that inherently has storage, but the description of Fig. 1 states that "Bitmaps are read into a **cell** from disk". Here Piersol implies that it's the **cell** (not an **object** instantiated in the cell) that has storage for receiving a bitmap. This is also supported by the text on page 387, col. 1, paragraph 4, line 1, i.e., "An image processing spreadsheet can read in images from disk, ...". Thus, Piersol does NOT disclose a spreadsheet that includes cells in which objects are instantiated, as required by claim 1 of Applicant's invention.

The Examiner asserts that "Piersol also discloses the objects as providing member functions that perform various functions on the object". It is further stated on page 387, col. 1, paragraph 4, line 2, "... then perform **any of several transformations**". This teaches away from Applicant's single method object (and method objects in general) because a single method object **includes** "a single member function, the single member function being adapted to access internal data stored in the

single method object and return a single value", as set forth in twice-amended claim 1. Also, see Applicant's specification, page 3, third paragraph, lines 3-4, and page 4, first paragraph, lines 1-2. By contrast, Piersol teaches that after the image processing spreadsheet reads in images from disk into **any** of the cells, and **any of several** **transformations** can be performed on the images in **any** of the cells. Thus, there is **no association** taught by Piersol of an object, internal storage, and a single "method", such as a single "transformation", as claimed in twice-amended claim 1. Consequently, Piersol does not disclose "the objects as providing member functions that perform various functions on the object".

The Examiner admits that "Piersol does not explicitly disclose a single method object". The Examiner then again cites pg 386, Fig. 1, and pg 387, col. 1, paragraph 4. However, these cites show that Piersol teaches away from Applicant's claimed "single method object", as explained above.

The Examiner also asserts that "Piersol also discloses a flexible and extensible analytic spreadsheet package (ASP) in which any new data types can be implemented immediately and to which it is easy enough for non-programmers to add new functions". There are several things wrong with this assertion.

First, there is nothing in Piersol that teaches how to implement a single method object of Applicant's invention as a "Smalltalk--80 object", and in fact, it may not even be possible to implement a single method object as a "Smalltalk--80 object", since Piersol requires that a new data type must first be implemented in Smalltalk (page 1, par 3, lines 1-2). Further restrictions exist, such as restrictions on Cell References in ASP Formulas. (page 386, paragraphs 1-3). For example, formulas are limited by the

Smalltalk compiler's inability to compile arbitrarily complex blocks. Moreover, certain constructions are forbidden, such as altering the value of a cell from within the rule of another cell. As stated by Piersol, "This appears to be a very difficult problem, given the present design of ASP and Smalltalk, and is probably unresolvable". (page 386, paragraph 3, lines 7-9)

Further, the Examiner's assertion that the statement that "any new data types can be implemented immediately and to which it is easy enough for non-programmers to add new functions" implies that the "single method objects" of Applicant's invention can trivially be created is mere speculation, and there is nothing in Piersol that teaches, suggests, or motivates the creation of Applicant's "single method objects". In fact, in addition to the restrictions recited in the previous paragraph herein, Piersol cautions that "no language is helpful on every sort of problem" (page 387, col. 2, paragraph 2, lines 3-4). If a teaching of any capability exists in Piersol to **create** the "single method object" of Applicant's invention, any teaching, suggestion, or motivation to create Applicant's invention using the scripting language of Piersol **does not teach enough to enable** Applicant's invention, since it would clearly require **undue experimentation** to provide Applicant's invention, given **solely** Piersol's disclosure. Consequently, to create a "single method object" as set forth in the first element of Applicant's claim 1 would require substantial inventiveness **in addition to** Piersol's disclosed teaching. Thus, it would NOT be obvious to one of ordinary skill in the art at the time of the invention to take advantage of the disclosed ASP in Piersol to implement the single method object of Applicant's invention.

The Examiner then goes on to speculate that "Such a modification might be taken advantage of in order to create a spreadsheet that is of specific use (e.g. by only implementing said single method object, thereby limiting the user's application to one of image processing)." However, this is not true, since single method objects are a general class of objects that do not limit a user to image processing methods. Further, it is not true that implementing "single method objects" would limit a user's application to image processing. In principle, one could perform a myriad of applications using "single method objects", since the use of single method objects does not impose limits on the type of data stored, nor on the type of methods associated with the data by the single method object.

The Examiner admits that Piersol does not explicitly disclose returning a **single** value, as required by the single method object of claim 1. The Examiner then speculates that "Piersol's disclosure of a flexible and extensible ASP as previously stated raises a case of obviousness". However, as stated above herein, there is nothing in Piersol that teaches, suggests, or motivates the creation of Applicant's "single method objects" including the return of a **single** value, using the ASP of Piersol or otherwise. In fact, Fig. 1 of Piersol teaches away from returning a single value, showing how "new bitmaps are created in other cells (objects) by sending Smalltalk-80 image manipulation messages", where each "new bitmap" is clearly a **plurality** of values, not a **single** value.

The Examiner asserts that "Piersol inherently discloses the return of **values** from objects by disclosing using functions of objects to create transformation of images stored in the objects." Indeed, Piersol teaches image processing (transformation) such

as illustrated by Fig. 1 of Piersol, wherein an image that consists of an array of pixel values is transformed into another array of pixel values. By contrast, Applicant's "single method object" returns a **single** value. Thus, Piersol teaches away from Applicant's invention.

The Examiner asserts that "it would have been obvious to one of ordinary skill in the art at the time of the invention to write new functions as disclosed in order to specify a return value of a member function." In fact, Piersol does not teach, suggest, or motivate the creation of "new functions" that return a single value. Piersol merely states that "the use of these features allow ASP to handle a very diverse range of data types, as well as a very broad range of operations that can be performed upon them" (page 385, col. 1, paragraph 3, lines 5-7), yet such broad statements fail to provide any teaching, suggestion, or motivation to create the "single method objects" that return a "single value", as taught and claimed by Applicant. It is the **Applicant that first recognized** the benefits of tailoring "a spreadsheet to a user's needs by specifying the functions of an object and their return values" within the context of the first element of Applicant's claim 1, for example, even though it now seems **obvious to the Examiner in hindsight** in view of Piersol and the art of object-oriented environments at the time of the invention.

The Examiner admits that "Piersol does not explicitly teach a data display buffer displaying its contents under a transparent grid". (Note that claims 1 and 2 have been amended to make it more clear that the spreadsheet claimed by Applicant is "partially transparent", as shown in Figs. 1 - 6 of Applicant's specification.) Applicant accordingly

asserts that Piersol does not teach a data display buffer displaying its contents under a partially transparent grid.

The Examiner asserts that "Levoy teaches a data display buffer that displays the data content of an object, the content being displayed in the buffer when the cell corresponding to the object is selected", citing page 6, Fig. 4; and page 1, col. 1, paragraph 3. However, Fig. 4 shows how "graphical objects" such as images ... are **displayed in miniature** inside each cell. Thus, **each cell** has an image display buffer associated with it. By contrast, Applicant's invention employs only a display buffer associated with the **spreadsheet grid** for display under the **spreadsheet grid**, as shown in Figs. 1-6, and claimed.

Further, double clicking on (selecting) a cell brings up a "full-size" object, as stated on page 1, col. 2, paragraph 3, lines 3-5. By contrast, graphical objects in Applicants invention are **not** visible to a user until he/she selects a cell, whereupon the image contained therein appears **below** a partially transparent spreadsheet that contains the selected cell, as required by claim 1.

Moreover, the "full size" object is **opaque**, thereby blocking at least part of the spreadsheet from view by a user since it is placed **above** the spreadsheet in Fig. 4. Likewise, the "Emacs" window also obscures the user's view of the spreadsheet below.

The Examiner then asserts that it would have been obvious to combine the teachings of Levoy with the invention disclosed by Piersol to provide a user with different viewing options, such as creating a separate display buffer containing a larger version of said image. Yet, combining the teaching of Levoy with the teaching of Piersol would not result in Applicant's invention in that both Levoy and Piersol teach a

spreadsheet that shows multiple images simultaneously, resulting in a complex and possibly confusing display. Levoy adds to the confusion by covering up some of the images with an opaque larger duplicate image and even an opaque Emacs window. By contrast, Applicant's spreadsheet shows only a selected image, all the other image data stored in other single method objects instantiated in other cells not being visible under the spreadsheet grid until being selected by a user. It is inherent in the invention of claim 1 that only a single image is displayed under the spreadsheet grid, since the second element of twice-amended claim 1 calls for "a data buffer, the data contents of which are displayed under a partially transparent spreadsheet grid". Thus, since combining Levoy and Piersol would result in a departure from Applicant's invention, it would not be obvious to combine them to obtain Applicant's invention.

The Examiner admits that "Piersol and Levoy do not explicitly disclose the data contents as being displayed under a transparent spreadsheet grid". The Examiner then asserts that "Smith teaches a graphical user interface in which portions of the display may be transparent at one time". In fact, in Smith, "transparent" means **invisible**, i.e., "not displayed" (col. 2, line 28). By contrast, Applicant's spreadsheet grid is "partially transparent", as set forth in twice-amended claim 1, and as shown in Figs. 1-6 of Applicant's specification.

Further, in Smith, the position of the pointer determines which portion of the grid becomes blocked by an opaque control, and which part becomes visible due to a transparent control. This feature shows that visibility of the grid is determined by the position of a cursor or pointer. (see, for example, col. 7, lines 40-52) By contrast,

Applicant's partially transparent grid is always visible, regardless of the position of the cursor, and regardless of which cell is selected.

Smith also teaches a "semi-transparent" grid at col. 8, lines 17-18, in cooperation with an opaque element 610 under a cursor 606, wherein the opaque element 610 covers a plurality of cells of the semi-transparent grid (see col. 8, lines 25-26), "such that only these entries are visible". Thus, the transparency of the grid of Smith changes with the position of the cursor, whereas in Applicant's invention, the transparency of the grid is independent of cursor position. Instead, in Applicant's invention, the position of the cursor over a cell selects that cell such that the image data (or any data set) stored in the single method object instantiated in that cell is displayed in the "data display buffer ... under a partially transparent spreadsheet grid", as set forth in twice-amended claim 1. Thus, the behavior of the semi-transparent grid of Smith is different from the behavior of the partially transparent grid of Applicant's invention.

Consequently, since the transparency taught by Smith depends on the position of the cursor, which is very different functionality from the functionality taught by Applicant, combining Smith with Piersol and Levoy cannot even possibly result in Applicant's invention. Further, as argued above, even if Smith provided the same partially transparent grid functionality as taught and claimed by Applicant, Smith does not remedy the deficiencies of the combination of Piersol and Levoy.

Moreover, even if the combination of Piersol, Levoy, and Smith did result in Applicant's invention, there is nothing taught, suggested, or motivated by any of these references that would indicate combining them.

Yet further, it's clear, as argued above, that **combining these references would NOT provide Applicant's invention.**

The Examiner asserts that "such a modification would have provided access of an **entire spreadsheet** to a user at an instant instead of covering parts. Yet, this statement is false because, as shown above, Smith teaches that a "semi-transparent" grid at col. 8, lines 17-18, in cooperation with an opaque element 610 under a cursor 606, wherein the opaque element 610 covers a plurality of cells of the semi-transparent grid (see col. 8, lines 25-26), "**such that only these entries are visible**". Thus, the modification taught by Smith does NOT provide access to the entire spreadsheet to a user at an instant.

Further, since Smith is **not providing even partial transparency**, instead allowing **a portion of the grid to be completely obscured by an opaque portion**, it is also not true that providing the "transparency" of Smith would allow "a user to search for desired data when the user was unsure if the data resided under the area covered by the data buffer", because Smith does not provide transparency over the entire area of the image displayed in the data buffer.

Accordingly, for any and all of the above-explained reasons, the rejection of claim 1 under 35 USC 103(a) is deemed to be overcome.

Regarding independent claim 2, the Examiner asserts that "Piersol discloses a spreadsheet that includes cells in which objects are instantiated", **but** at page 385, col. 2, paragraph 2, lines 5-8, it's clear that Piersol merely discloses that "images ... have been included in ASP cells". An **image** included in an ASP cell is NOT equivalent to a

single method object instantiated in a cell of Applicant's invention. Moreover, a **cell** is NOT equivalent to a **single method object** of Applicant's invention, and therefore an image included in an ASP cell is not equivalent to an image included in a single method object.

In fact, Piersol states that "Each cell in an ASP spreadsheet is an object of class SpreadsheetCell. By making each cell a separate object, operations on the cells themselves (as opposed to cell values) are simplified." (page 388, col. 1, paragraph 1, lines 1-4) Thus, in Piersol, **images are merely stored in objects**, i.e., the **cells** of Piersol's spreadsheet, whereas in Applicant's invention, **images are first stored in respective "single method objects"**, that are in turn instantiated in at least one cell of Applicant's spreadsheet.

The Examiner cites Fig. 1 of Piersol as showing an example of an object that inherently has storage, but the description of Fig. 1 states that "Bitmaps are read into a **cell** from disk". Here Piersol implies that it's the **cell** (not an **object** instantiated in the cell) that has storage for receiving a bitmap. This is also supported by the text on page 387, col. 1, paragraph 4, line 1, i.e., "An image processing spreadsheet can read in images from disk, ...". Thus, Piersol does NOT disclose a spreadsheet that includes cells in which objects are instantiated, as required by claim 2 of Applicant's invention.

The Examiner admits that "Piersol does not explicitly disclose a **single method object**." The Examiner then asserts that Piersol does disclose objects that act as data buffering functions, citing pg 386, Fig. 1, and pag 387, col. 1, par 4). First, a single method object does more than buffer data; there must also be associated with the data a **single method** adapted to access the data and return a single value.

It is further stated on page 387, col. 1, paragraph 4, line 2, “... then perform **any of several transformations**”. This teaches away from Applicant’s **single method object** (and method objects in general) because a single method object **includes** “a member function, the member function being adapted to access the single method object and return a single value”, as set forth in twice-amended claim 2. Also, see Applicant’s specification, page 3, third paragraph, lines 3-4, and page 4, first paragraph, lines 1-2.

By contrast, Piersol teaches that after the image processing spreadsheet reads in images from disk into **any of the cells**, and **any of several transformations** can be performed on the images in **any of the cells**. Thus, there is **no association** taught by Piersol of an object, internal storage, and a single “method”, such as a single “transformation”, as claimed in twice-amended claim 2. Consequently, Piersol does not disclose the first element of twice-amended claim 2, i.e., “instantiating a single method object in each of a plurality of the cells of the spreadsheet, each single method object being adapted to provide internal storage for storing a large data set, and a single member function adapted to access the large data set and return a single value”.

The Examiner also asserts that “Piersol also discloses a flexible and extensible analytic spreadsheet package (ASP) in which any new data types can be implemented immediately and to which it is easy enough for non-programmers to add new functions”. There are several things wrong with this assertion.

First, there is nothing in Piersol that teaches how to implement a single method object of Applicant’s invention as a “Smalltalk-80 object”, and in fact, it may not even be possible to implement a single method object as a “Smalltalk-80 object”, since Piersol

requires that a new data type must first be implemented in Smalltalk (page 1, par 3, lines 1-2). Further restrictions exist, such as restrictions on Cell References in ASP Formulas. (page 386, paragraphs 1-3). For example, formulas are limited by the Smalltalk compiler's inability to compile arbitrarily complex blocks. Moreover, certain constructions are forbidden, such as altering the value of a cell from within the rule of another cell. As stated by Piersol, "This appears to be a very difficult problem, given the present design of ASP and Smalltalk, and is probably unresolvable". (page 386, paragraph 3, lines 7-9)

In fact, in addition to the restrictions recited in the previous paragraph herein, Piersol cautions that "no language is helpful on every sort of problem" (page 387, col. 2, paragraph 2, lines 3-4). If a teaching of any capability exists in Piersol to **create** the "single method object" of Applicant's invention, any teaching, suggestion, or motivation to create Applicant's invention using the scripting language of Piersol **does not teach enough to enable** Applicant's invention, since it would clearly require **undue experimentation** to provide Applicant's invention, given **solely** Piersol's disclosure. Consequently, to create a "single method object" as set forth in the first element of Applicant's claim 2 would require substantial inventiveness **in addition to** Piersol's disclosed teaching. Thus, it would NOT be obvious to one of ordinary skill in the art at the time of the invention to take advantage of the disclosed ASP in Piersol to implement the single method object of Applicant's invention.

The Examiner then goes on to speculate that "Such a modification might be taken advantage of in order to create a spreadsheet that is of specific use (e.g. by only implementing said single method object, thereby limiting the user's application to one of

image processing)." However, this is not true, since single method objects are a general class of objects that do not limit a user to image processing methods. Consequently, it is not true that implementing "single method objects" would limit a user's application to image processing. In principle, one could perform a myriad of applications using "single method objects", since the use single method objects does not impose limits on the type of data stored, nor on the type of methods associated with the data by the single method object.

In view of the amendment to claim two requiring that a single method object "return a single value", the Examiner might admit as he had in regard to claim 1 that Piersol does not explicitly disclose returning a **single** value, as required by the single method object of claim 2. The Examiner then would probably also again speculate that "Piersol's disclosure of a flexible and extensible ASP as previously stated raises a case of obviousness". However, as stated above herein, there is nothing in Piersol that teaches, suggests, or motivates the creation of Applicant's "single method objects" including the return of a **single** value, using the ASP of Piersol or otherwise.

The Examiner then might analogously assert that "Piersol inherently discloses the return of **values** from objects by disclosing using functions of objects to create transformation of images stored in the objects." Indeed, Piersol teaches image processing (transformation) such as illustrated by Fig. 1 of Piersol, wherein an image that consists of an array of pixel values is transformed into another array of pixel values. By contrast, Applicant's "single method object" returns a **single** value. Thus, Piersol teaches away from Applicant's invention.

In view of the amendment to claim 2, the Examiner might also analogously assert that "it would have been obvious to one of ordinary skill in the art at the time of the invention to write new functions as disclosed in order to specify a return value of a member function." In fact, Piersol does not teach, suggest, or motivate the creation of "new functions" that return a single value. Piersol merely states that "the use of these features allow ASP to handle a very diverse range of data types, as well as a very broad range of operations that can be performed upon them" (page 385, col. 1, paragraph 3, lines 5-7), yet such broad statements fail to provide any teaching, suggestion, or motivation to create the "single method objects" that return a "single value", as taught and claimed by Applicant. It is the **Applicant that first recognized** the benefits of tailoring "a spreadsheet to a user's needs by specifying the functions of an object and their return values" within the context of the first element of Applicant's twice-amended claim 2, for example, even though it might seem **obvious to the Examiner in hindsight** in view of Piersol and the art of object-oriented environments at the time of the invention.

The Examiner admits that "Piersol does not explicitly teach displaying a large data set corresponding to a selected cell of a spreadsheet". The Examiner then asserts that "Levoy teaches displaying an image stored in an object when the cell corresponding to the object is selected", citing fig. 4 on page 6, and page 1, col. 1, paragraph 3.

However, Fig. 4 shows how "graphical objects" such as images ... are **displayed in miniature** inside each cell. Thus, **each cell** in Peirsol has an image display buffer associated with it. By contrast, Applicant's invention includes "displaying the large data

set of a single method object corresponding to a **selected cell** of the spreadsheet in which the single method object is instantiated", as shown in Figs. 1-6, and claimed in twice-amended claim 2. Note that in Applicant's invention, all cells other than the selected cell are not associated with an image display buffer, and do not display an image of the data stored in a cell instantiated therein.

Further, double clicking on (selecting) a cell **brings up** over the grid a "full-size" object, as stated on page 1, col. 2, paragraph 3, lines 3-5 that obscures the underlying grid, as shown in Fig. 4. By contrast, graphical objects in Applicants invention are **not** visible to a user until he/she selects a cell, whereupon the image contained therein appears **below** a partially transparent spreadsheet grid that contains the selected cell, as required by claim 2.

Note well that the "full size" object is **opaque**, thereby blocking at least part of the spreadsheet from view by a user since it is placed **above** the spreadsheet in Fig. 4. Likewise, the "Emacs" window also obscures the user's view of the spreadsheet below.

The Examiner then asserts that it would have been obvious to combine the teachings of Levoy with the invention disclosed by Piersol to provide a user with different viewing options, **such as creating a separate display containing a larger version of said image.**

Yet, combining the teaching of Levoy with the teaching of Piersol would not result in Applicant's invention in that both Levoy and Piersol teach a spreadsheet that shows multiple images simultaneously, resulting in a complex and possibly confusing display. Levoy adds to the confusion by covering up some of the images with an opaque larger duplicate image and even an opaque Emacs window. By contrast,

Applicant's spreadsheet shows only a selected image, all the other data stored in other single method objects instantiated in other cells not being visible under the spreadsheet grid until being selected by a user. It is inherent in the invention of claim 2 that only a single image (large data set) is displayed under the partially transparent spreadsheet, since the third element of twice-amended claim 2 calls for "displaying in superimposed relationship with the large data set a partially transparent spreadsheet including the selected cell". Thus, since combining Levoy and Piersol would result in a departure from Applicant's invention, it would not be advisable, or obvious, to combine them.

The Examiner admits that "Piersol and Levoy do not explicitly disclose the data contents as being displayed under a transparent spreadsheet grid". The Examiner then asserts that "Smith teaches a graphical user interface in which portion of the display may be transparent at one time". In fact, in Smith, "transparent" means **invisible**, i.e., "not displayed" (col. 2, line 28). By contrast, Applicant's spreadsheet grid is "partially transparent", as set forth in twice-amended claim 2.

Further, in Smith, the position of the pointer determines which portion of the grid becomes blocked by an opaque control, and which part becomes visible due to a transparent control. This feature shows that visibility of the grid is determined by the position of a cursor or pointer. (see, for example, col. 7, lines 40-52) By contrast, Applicant's partially transparent grid is always visible, regardless of the position of the cursor, and regardless of which cell is selected.

Smith also teaches a "semi-transparent" grid at col. 8, lines 17-18, in cooperation with an opaque element 610 under a cursor 606, wherein the opaque element 610 covers a plurality of cells of the semi-transparent grid (see col. 8, lines 25-26), "such

that only these entries are visible". Thus, the transparency of the grid of Smith changes with the position of the cursor, whereas in Applicant's invention, the transparency of the grid is independent of cursor position. Instead, in Applicant's invention, the position of the cursor over a cell selects that cell such that the image data stored in the single method object instantiated in that cell is displayed in the data buffer under the partially transparent spreadsheet grid, as set forth in twice-amended claim 2. Thus, the behavior of the semi-transparent grid of Smith is different from the behavior of the partially transparent grid of Applicant's invention.

Consequently, since the transparency taught by Smith depends on the position of the cursor, which is very different functionality from the functionality taught by Applicant, combining Smith with Piersol and Levoy cannot even possibly result in Applicant's invention. Further, as argued above, even if Smith provided the same partially transparent grid functionality as taught and claimed by Applicant, Smith does not remedy the deficiencies of the combination of Piersol and Levoy.

Moreover, even if the combination of Piersol, Levoy, and Smith did result in Applicant's invention, there is nothing taught, suggested, or motivated by any of these references that would indicate combining them.

Yet further, it's clear, as argued above, that **combining these references would NOT provide Applicant's invention.**

The Examiner asserts that "such a modification would have provided access of an **entire** spreadsheet to a user at an instant instead of covering parts. Yet, this statement is false because, as shown above, Smith teaches that a "semi-transparent" grid at col. 8, lines 17-18, in cooperation with an opaque element 610 under a cursor

606, wherein the opaque element 610 covers a plurality of cells of the semi-transparent grid (see col. 8, lines 25-26), "**such that only these entries are visible**". Thus, the modification taught by Smith does NOT provide access to the "entire spreadsheet to a user at an instant, instead of covering parts", as asserted by the Examiner.

Further, since Smith is **not providing even partial transparency**, instead **allowing a portion of the grid to be completely obscured by an opaque portion**, it is also not true that providing the "transparency" of Smith would allow "a user to search for desired data when the user was unsure if the data resided under the area covered by the data buffer", because Smith does not provide transparency over the entire area of the image displayed in the data buffer.

Accordingly, for any and all of the above-explained reasons, the rejection of claim 2 under 35 USC 103(a) is deemed to be overcome.

Regarding independent claim 3, the Examiner asserts that "Piersol discloses a spreadsheet that includes cells in which objects are instantiated", **but at page 385, col. 2, paragraph 2, lines 5-8, it's clear that Piersol merely discloses that "images.... have been included in ASP cells". An image included in an ASP cell is NOT equivalent to a single method object instantiated in a cell of Applicant's invention. Moreover, a cell is NOT equivalent to a single method object of Applicant's invention, and therefore an image included in an ASP cell is not equivalent to an image included in a single method object.**

In fact, Piersol states that "Each cell in an ASP spreadsheet is an object of class SpreadsheetCell. By making each cell a separate object, operations on the cells

themselves (as opposed to cell values) are simplified." (page 388, col. 1, paragraph 1, lines 1-4) Thus, in Piersol, **images are merely stored in objects, i.e., the cells of** Piersol's spreadsheet, whereas in Applicant's invention, **images are first stored in respective "single method objects", that are in turn instantiated in at least one** cell of Applicant's spreadsheet.

The Examiner cites Fig. 1 of Piersol as showing an example of an object that inherently has storage, but the description of Fig. 1 states that "Bitmaps are read into a **cell from disk**". Here Piersol implies that it's the **cell** (not an **object** instantiated in the cell) that has storage for receiving a bitmap. This is also supported by the text on page 387, col. 1, paragraph 4, line 1, i.e., "An image processing spreadsheet can read in images from disk, ...". Thus, Piersol does NOT disclose a spreadsheet that includes cells in which objects are instantiated, as required by claim 3 of Applicant's invention.

The Examiner admits that "Piersol does not explicitly disclose a **single method object**." The Examiner then asserts that Piersol does disclose objects that act as data buffering functions, citing pg 386, Fig. 1, and pag 387, col. 1, par 4). First, a single method object does more than buffer data; there must also be associated with the data a **single method** adapted to access the data and return a single value.

It is further stated on page 387, col. 1, paragraph 4, line 2, " ... then perform **any of several transformations**". This teaches away from Applicant's **single method object** (and method objects in general) because a single method object **includes** "a single member function, the single member function being adapted to access the single method object and return a single value", as set forth in amended claim 3. Also, see

Applicant's specification, page 3, third paragraph, lines 3-4, and page 4, first paragraph, lines 1-2.

By contrast, Piersol teaches that after the image processing spreadsheet reads in images from disk into **any** of the cells, and **any of several transformations** can be performed on the images in **any** of the cells. Thus, there is **no association** taught by Piersol of an object, internal storage, and a single "method", such as a single "transformation", as claimed in amended claim 3. Consequently, Piersol does not disclose the first element of amended claim 3, i.e., "instantiating a single method object in each of a plurality of the cells of the spreadsheet, each single method object being adapted to provide internal storage for storing a large data set, and a single member function adapted to access the large data set and return a single value".

The Examiner also asserts that "Piersol also discloses a flexible and extensible analytic spreadsheet package (ASP) in which any new data types can be implemented immediately and to which it is easy enough for non-programmers to add new functions". There are several things wrong with this assertion.

First, there is nothing in Piersol that teaches how to implement a single method object of Applicant's invention as a "Smalltalk-80 object", and in fact, it may not even be possible to implement a single method object as a "Smalltalk-80 object", since Piersol requires that a new data type must first be implemented in Smalltalk (page 1, par 3, lines 1-2). Further restrictions exist, such as restrictions on Cell References in ASP Formulas. (page 386, paragraphs 1-3). For example, formulas are limited by the Smalltalk compiler's inability to compile arbitrarily complex blocks. Moreover, certain constructions are forbidden, such as altering the value of a cell from within the rule of

another cell. As stated by Piersol, "This appears to be a very difficult problem, given the present design of ASP and Smalltalk, and is probably unresolvable". (page 386, paragraph 3, lines 7-9)

In fact, in addition to the restrictions recited in the previous paragraph herein, Piersol cautions that "no language is helpful on every sort of problem" (page 387, col. 2, paragraph 2, lines 3-4). If a teaching of any capability exists in Piersol to **create** the "single method object" of Applicant's invention, any teaching, suggestion, or motivation to create Applicant's invention using the scripting language of Piersol **does not teach enough to enable** Applicant's invention, since it would clearly require **undue experimentation** to provide Applicant's invention, given **solely** Piersol's disclosure. Consequently, to create a "single method object" as set forth in the first element of Applicant's claim 3 would require substantial inventiveness **in addition to** Piersol's disclosed teaching. Thus, it would NOT be obvious to one of ordinary skill in the art at the time of the invention to take advantage of the disclosed ASP in Piersol to implement the single method object of Applicant's invention.

The Examiner then goes on to speculate that "Such a modification might be taken advantage of in order to create a spreadsheet that is of specific use (e.g. by only implementing said single method object, thereby limiting the user's application to one of image processing)." However, this is not true, since single method objects are a general class of objects that do not limit a user to image processing methods. Consequently, it is not true that implementing "single method objects" would limit a user's application to image processing. In principle, one could perform a myriad of applications using "single method objects", since the use single method objects does

not impose limits on the type of data stored, nor on the type of methods associated with the data by the single method object.

In view of the amendment to claim two requiring that a single method object "return a single value", the Examiner might admit as he had in regard to claim 1 that Piersol does not explicitly disclose returning a **single** value, as required by the single method object of claim 3. The Examiner then would probably also again speculate that "Piersol's disclosure of a flexible and extensible ASP as previously stated raises a case of obviousness". However, as stated above herein, there is nothing in Piersol that teaches, suggests, or motivates the creation of Applicant's "single method objects" including the return of a **single** value, using the ASP of Piersol or otherwise.

The Examiner then might analogously assert that "Piersol inherently discloses the return of **values** from objects by disclosing using functions of objects to create transformation of images stored in the objects." Indeed, Piersol teaches image processing (transformation) such as illustrated by Fig. 1 of Piersol, wherein an image that consists of an array of pixel values is transformed into another array of pixel values. By contrast, Applicant's "single method object" returns a **single** value. Thus, Piersol teaches away from Applicant's invention.

In view of the amendment to claim 3, the Examiner might also analogously assert that "it would have been obvious to one of ordinary skill in the art at the time of the invention to write new functions as disclosed in order to specify a return value of a member function." In fact, Piersol does not teach, suggest, or motivate the creation of "new functions" that return a single value. Piersol merely states that "the use of these features allow ASP to handle a very diverse range of data types, as well as a very broad

range of operations that can be performed upon them" (page 385, col. 1, paragraph 3, lines 5-7), yet such broad statements fail to provide any teaching, suggestion, or motivation to create the "single method objects" that return a "single value", as taught and claimed by Applicant. It is the **Applicant that first recognized** the benefits of tailoring "a spreadsheet to a user's needs by specifying the functions of an object and their return values" within the context of the first element of Applicant's amended claim 3, for example, even though it might seem **obvious to the Examiner in hindsight** in view of Piersol and the art of object-oriented environments at the time of the invention.

The Examiner admits that "Piersol does not explicitly teach selecting a cell and then displaying the contents stored in the object corresponding to said selected cell". The Examiner then asserts that "Levoy teaches displaying an image stored in an object when the cell corresponding to the object is selected", citing fig. 4 on page 6, and page 1, col. 1, paragraph 3.

However, Fig. 4 shows how "graphical objects" such as images ... are **displayed in miniature** inside each cell. Thus, **each cell** in Peirsol has an image display buffer associated with it. By contrast, Applicant's invention includes "displaying the large data set of a single method object corresponding to a **selected cell** of the spreadsheet in which the single method object is instantiated", as shown in Figs. 1-6, and claimed in amended claim 3. Note that in Applicant's invention, all cells other than the selected cell are not associated with an image display buffer, and do not display an image of the data stored in a cell instantiated therein.

Further, double clicking on (selecting) a cell **brings up** over the grid a "full-size" object, as stated on page 1, col. 2, paragraph 3, lines 3-5 that obscures the underlying

grid, as shown in Fig. 4. By contrast, graphical objects in Applicants invention are **not** visible to a user until he/she selects a cell, whereupon the image contained therein appears **below** a partially transparent spreadsheet grid that contains the selected cell, as required by claim 3.

Note well that the “full size” object is **opaque**, thereby blocking at least part of the spreadsheet from view by a user since it is placed **above** the spreadsheet in Fig. 4. Likewise, the “Emacs” window also obscures the user’s view of the spreadsheet below.

The Examiner then asserts that it would have been obvious to combine the teachings of Levoy with the invention disclosed by Piersol to provide a user with different viewing options, **such as creating a separate display containing a larger version of said image.**

Yet, combining the teaching of Levoy with the teaching of Piersol would not result in Applicant’s invention in that both Levoy and Piersol teach a spreadsheet that shows multiple images simultaneously, resulting in a complex and possibly confusing display. Levoy adds to the confusion by covering up some of the images with an opaque larger duplicate image and even an opaque Emacs window. By contrast, Applicant’s spreadsheet shows only a selected image, all the other data stored in other single method objects instantiated in other cells not being visible under the spreadsheet grid until being selected by a user. It is inherent in the invention of claim 3 that only a single image (large data set) is displayed under the partially transparent spreadsheet, since the third element of amended claim 3 calls for “displaying in superimposed relationship with the large data set a partially transparent spreadsheet including the

selected cell". Thus, since combining Levoy and Piersol would result in a departure from Applicant's invention, it would not be advisable, or obvious, to combine them.

The Examiner admits that "Piersol and Levoy do not explicitly disclose an image as being displayed in superimposed relationship with a transparent spreadsheet". The Examiner then asserts that "Smith teaches a graphical user interface in which portion of the display may be transparent at one time". In fact, in Smith, "transparent" means **invisible**, i.e., "not displayed" (col. 2, line 28). By contrast, Applicant's spreadsheet grid is "partially transparent", as set forth in amended claim 3.

Further, in Smith, the position of the pointer determines which portion of the grid becomes blocked by an opaque control, and which part becomes visible due to a transparent control. This feature shows that visibility of the grid is determined by the position of a cursor or pointer. (see, for example, col. 7, lines 40-52) By contrast, Applicant's partially transparent grid is always visible, regardless of the position of the cursor, and regardless of which cell is selected.

Smith also teaches a "semi-transparent" grid at col. 8, lines 17-18, in cooperation with an opaque element 610 under a cursor 606, wherein the opaque element 610 covers a plurality of cells of the semi-transparent grid (see col. 8, lines 25-26), "such that only these entries are visible". Thus, the transparency of the grid of Smith changes with the position of the cursor, whereas in Applicant's invention, the transparency of the grid is independent of cursor position. Instead, in Applicant's invention, the position of the cursor over a cell selects that cell such that the image data stored in the single method object instantiated in that cell is displayed in the data buffer under the partially transparent spreadsheet grid, as set forth in amended claim 3. Thus, the behavior of

the semi-transparent grid of Smith is different from the behavior of the partially transparent grid of Applicant's invention.

Consequently, since the transparency taught by Smith depends on the position of the cursor, which is very different functionality from the functionality taught by Applicant, combining Smith with Piersol and Levoy cannot even possibly result in Applicant's invention. Further, as argued above, even if Smith provided the same partially transparent grid functionality as taught and claimed by Applicant, Smith does not remedy the deficiencies of the combination of Piersol and Levoy.

Moreover, even if the combination of Piersol, Levoy, and Smith did result in Applicant's invention, there is nothing taught, suggested, or motivated by any of these references that would indicate combining them.

Yet further, it's clear, as argued above, that **combining these references would NOT provide Applicant's invention.**

The Examiner asserts that "such a modification would have provided access of an **entire** spreadsheet to a user at an instant instead of covering parts. Yet, this statement is false because, as shown above, Smith teaches that a "semi-transparent" grid at col. 8, lines 17-18, in cooperation with an opaque element 610 under a cursor 606, wherein the opaque element 610 covers a plurality of cells of the semi-transparent grid (see col. 8, lines 25-26), "**such that only these entries are visible**". Thus, the modification taught by Smith does NOT provide access to the "entire spreadsheet to a user at an instant, instead of covering parts", as asserted by the Examiner.

Further, since Smith is **not providing even partial transparency**, instead allowing a **portion of the grid to be completely obscured by an opaque portion**, it is

also not true that providing the "transparency" of Smith would allow "a user to search for desired data when the user was unsure if the data resided under the area covered by the data buffer", because Smith does not provide transparency over the entire area of the image displayed in the data buffer.

Accordingly, for any and all of the above-explained reasons, the rejection of claim 3 under 35 USC 103(a) is deemed to be overcome.

Regarding dependent claim 4, it has been amended to be consistent with claims 1, 2, and 3 in that the electronic spreadsheet is made explicitly **partially** transparent. The Examiner points out that "Smith teaches control that change between states of opaqueness according to a user's selection (e.g. completely opaque to (sic.) completely transparent; col. 2, lines 15-28)." However, it must be noted that the cited text illustrates that Smith toggles between two configurations, a first configuration having three controls that are opaque (col. 2, lines 18-19), and a second configuration having only one of the controls being opaque, the rest being invisible (wholly transparent) (col. 2, lines 20-22). There is no intermediate that could be described as **partially** transparent, as claimed in the herein-amended claims 1-4, where the partial transparency is **adjustable**, as required by amended claim 4. Instead, the transparency of Smith is more accurately described as **switchable**, as stated by Smith himself, i.e., "The particular event causing the controls to switch between these configurations ..." at col. 2, lines 22-23.

Additionally and alternatively, claim 4 depends from claim 3 which is deemed to be allowable since combining Piersol, Levoy, and Smith would not provide the invention of claim 3, nor is there any teaching in Piersol, Levoy, and Smith that would teach,

suggest, or motivate such a combination, as demonstrated above. Consequently, claim 4 is deemed to be allowable, and the rejection of claim 4 under 35 USC 103(a) is deemed to be overcome.

Regarding dependent claim 5, the Examiner admits that “Smith does not explicitly disclose the use of a game controller.” The Examiner then asserts that “Smith teaches the use of mouses, touch pads, trackballs, remote controls, and point sticks as input devices.” However, a game controller does NOT provide the same functionality as mouses, touch pads, trackballs, remote controls, and point sticks, in that **a game controller has more degrees of freedom** than mouses, touch pads, trackballs, remote controls, and point sticks, each of which can only point and click. By contrast, a game controller has more input means than just means for pointing and clicking so as to facilitate more effective manipulation of the features of the advanced electronic spreadsheet of Applicant’s invention. It’s clear that Smith is teaching away from using a game controller because each of the many listed input devices can merely point and click.

The Examiner asserts that “It was common and typical in the art at the time of the invention to include game controllers among common input devices, yet neither Piersol, Levoy, and Smith teach using a game controller. If the “art” is the art of electronic spreadsheets, Applicant’s attorney is not aware of any reference that teaches using a game controller with an electronic spreadsheet, so it is surely not common and typical in the art.

The Examiner points out that "A game controller may have been a type of combination of remote controls, point sticks and mice", yet this is not true, nor is such a combination taught, suggested, or motivated by Piersol, Levoy, and Smith, alone or in any combination.

Further, Applicant's use of a game controller is NOT a standard computer input and pointing device for a spreadsheet, which is usually a tool for business and accounting. Game controllers are standard input and pointing devices for **computer games and video games**. However, keyboard and mouse are the standard input devices for spreadsheets. A game controller is clearly NOT a standard input and pointing device with respect to a spreadsheet. Thus, it is not obvious to use a game controller (usually used for amusement applications) with a spreadsheet (usually used for business applications). In fact, since game controllers are usually used for amusement applications such as video games, the art of video games is teaching away from the use of game controllers with spreadsheets.

In addition, or in the alternative, claim 5 depends upon claim 3, deemed allowable as explained above. Accordingly, the rejection of claim 5 under 35 USC 103(a) is deemed to be overcome.

Regarding dependent claim 6, this claim requires that "the selected cell is selected using one of a standard keyboard and a mouse". Smith teaches the use of keyboard and mouse, but does not teach, suggest, or motivate Applicant's invention, alone or in combination with Piersol and/or Levoy. Claim 6 depends upon claim 3,

deemed allowable as explained above. Accordingly, the rejection of claim 6 under 35 USC 103(a) is deemed to be overcome.

Regarding dependent claim 7, the Examiner asserts that "Smith teaches the superposition of an object image with a display of a graphical analysis of an object image", as required by claim 7, and cites Fig. 4 of Smith. However, Fig. 4 of Smith merely shows "operations of controls of a computer program displayable within a graphical user interface". Smith fails to show an "object image", as shown in Fig. 5 of Applicant's specification, and Smith fails to show a "graphical analysis of an object image" as also shown in Fig. 5 of Applicant's specification.

In addition, or in the alternative, claim 7 depends upon claim 3, deemed allowable as explained above. Accordingly, the rejection of claim 7 under 35 USC 103(a) is deemed to be overcome.

Claim 8 has been rejected under 35 USC 103(a) as being unpatentable over Piersol in view of Levoy and Smith, and further in view of Mastering Excel 97 4th ed. (herein "Excel"). Claim 8 requires that "the analysis of the object image is a histogram of the object image". As admitted by the Examiner, neither Piersol, Levoy nor Smith "explicitly disclose creating a histogram analysis". However, Excel does show using histograms for graphical analysis of data. Nevertheless, the histograms are **not superimposed with any other image, and in fact, are opaque**. For example, Fig. 13.2 shows a histogram or bar chart that actually is opaque, thereby blocking the user's view of the spreadsheet. This clearly **teaches away** from Applicant's invention as set

forth in claim 3, for example, that requires "displaying in superimposed relationship with the machine vision image a partially transparent electronic spreadsheet including the selected cell". Thus, Excel fails to show a transparent spreadsheet, as well as a "superposition of an object image and a graphical representation of an analysis of the object image", as required by claim 7, just as Piersol and Levoy fail. Moreover, Smith teaches transparent views of grids having opaque selected cells (col. 8, lines 15-20), but also fails to show a transparent grid cooperative with the superposition of images. None of the references show a partially transparent spreadsheet superimposed upon any image, whether a single or a superimposed image. Thus, combining Piersol, Levoy and Smith with Excel fails to provide Applicant's invention. Moreover, there is nothing in any of these references, taken together or alone, that teaches, suggests, or motivates combining them in any event.

In addition, or in the alternative, claim 8 depends upon claim 7, which depends from claim 3, deemed allowable as explained above. Accordingly, the rejection of claim 8 under 35 USC 103(a) is deemed to be overcome.

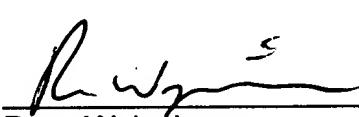
The prior art made of record and not relied upon has been reviewed and does not appear to present an impediment to allowance of the present application.

Attached hereto is a marked-up version of the changes made to the specification and claims by the current amendment. The attached page is captioned "VERSION WITH MARKINGS TO SHOW CHANGES MADE".

Accordingly, Applicants assert that the present application is in condition for allowance, and such action is respectfully requested. The Examiner is invited to phone the undersigned attorney to further the prosecution of the present application.

Respectfully Submitted,

Dated: 5/28/03



Russ Weinzimmer
Registration No. 36,717
Attorney for Applicant
P.O. Box 862
Wilton, NH 03086
Phone: 603-654-3524
Fax: 603-654-3556

VERSION WITH MARKINGS TO SHOW CHANGES MADE

In the claims:

Claims 1 - 4 have been amended as follows:

1. (Twice Amended) An electronic spreadsheet having a plurality of cells, the improvement comprising:

a single method object, adapted to be instantiated in at least one of the spreadsheet cells, and adapted to provide internal data storage and a single member function[s], the single member function[s] being adapted to access [the] internal data stored in the single method object and return a single value;

a data display buffer, the data contents of which are displayed under a partially transparent spreadsheet grid; and

means for selectively displaying the internal data [content] of the single method object in the data display buffer [, the single method object corresponding to a selected] by selecting a spreadsheet cell in which the single method object is instantiated.

2. (Twice Amended) A method for selectively displaying large data sets in an electronic spreadsheet having a plurality of cells, the method comprising:

instantiating a single method object in each of a plurality of the cells of the spreadsheet, each single method object being adapted to provide internal storage for storing a large data set, and a single member function adapted to access the large data set and return a single value;

displaying the large data set of the single method object corresponding to a selected cell of the spreadsheet in which the single method object is instantiated;
and

displaying in superimposed relationship with the large data set a partially transparent spreadsheet including the selected cell.

3. (Amended) A user-interface method for selectively displaying machine vision images stored in an electronic spreadsheet having a plurality of cells, the method comprising:

instantiating a single method object in each of a plurality of the cells of the spreadsheet, each single method object being adapted to provide internal storage for storing a machine vision image, and a single member function adapted to access the single method object and return a single value;

selecting a cell from the plurality of cells;
displaying the machine vision image stored in the single method object corresponding to the selected cell; and

displaying in superimposed relationship with the machine vision image a partially transparent electronic spreadsheet including the selected cell.

4. (Amended) The user-interface method of claim 3, wherein the partially transparent electronic spreadsheet is adjustably transparent.